Docket No.: 42390.P9429

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:
Jin Yang

Application No. 09/608,637

Filed: June 30, 2000

For: METHODS FOR FORMAL VERIFICATION ON A SYMBOLIC LATTICE DOMAIN

Examiner: E. Garcia Otero

Art Unit: 2123

CERTIFICATE OF TRANSMISSION

I hereby certify that this correspondence is being transmitted to the United States Patent and Trademark Office, or facsimile transmitted to Fax No. (531)273-8300

on 6-11-2007 Lawrence M. Mennemeier

Date Lawrence M. Mennemeier

APPELLANT'S BRIEF UNDER 37 CFR § 41.37 IN SUPPORT OF APPELLANT'S APPEAL TO THE BOARD OF PATENT APPEALS AND INTERFERENCES

Mail Stop Appeal Brief-Patents
Commissioner of Patents
PO Box 1450
Alexandria, VA 22313-1450

Dear Sir:

Appellant hereby submits this Brief in support of an appeal from a non-final decision of the Examiner, in the above-referenced case. Appellant respectfully requests consideration of this appeal by the board of Patent Appeals and Interference for allowance of the above-referenced patent application.

TABLE OF CONTENTS

I. Real Party in Interest

The real party in interest in the present appeal is Intel Corporation of Santa Clara, California, the assignee of the present application.

II. Related Appeals and Interferences

There are no related appeals or interferences to appellant's knowledge that would have a bearing on any decision of the Board of Patent Appeals and Interferences.

III. Status of the Claims (independent claims shown in bold)

Claims **1**-3, 6-7, **9**-13, **19**-20, **21**, **22**-27 and **29**-30 are canceled.

Claims **4**-5, **8**, **14**-15, **16**-18, **28** and 31-40 stand rejected under 35 USC § 112 as allegedly being indefinite and omitting essential structural relationships of elements.

Claims **4**-5, **8**, **14**-15, **16**-18, **28** and 31-40 stand rejected under 35 USC § 101 as allegedly lacking patentable utility.

Non-final rejection of claims **4**-5, **8**, **14**-15, **16**-18, **28** and 31-40 is being appealed.

IV. Status of Amendments

An official amendment and response to a first Office Action mailed 3/9/2004 was submitted by appellant on 9/9/2004 and was entered. A Final Office Action was mailed on 12/17/2004. Appellant responded by submitting an amendment and official response after final on 4/18/2005, which was entered. A Notice of Appeal was transmitted on 5/17/2005. An appeal brief and an amendment, under 37 CFR § 41.33 were submitted on 7/18/2005 and were entered.

A Non-final Office Action reopening prosecution and setting forth new grounds of rejection was mailed on 10/5/2005. An official amendment and response was submitted by appellant on 4/3/2006 and was entered. A Final Office Action was mailed on 6/30/2006. A Request for Continued Examination and response under 37 CFR § 1.114 were submitted on 9/28/2006 and were entered. A Non-final Office Action was mailed on 12/7/2006. A second Notice of Appeal was transmitted on 4/9/2007, and an appeal ensued.

Accordingly, the claims stand as of appelant's response of 9/28/2006, and are reproduced in clean form in the Claims Appendix.

V. Summary of Claimed Subject Matter

Appellant's disclosure describes methods for formal verification of circuits and other finite-state systems. Formal definitions and semantics are disclosed for a model of a finite-state system, an assertion graph to express properties for verification, and satisfiability criteria for specification and automated verification of forward implication properties and backward justification properties, the latter of which were formerly not supported through prior verification techniques. A method is also disclosed to compute a simulation relation sequence ending with a simulation relation fixpoint, which can be compared to a consequence labeling for each edge of an assertion graph to verify implication properties and justification properties according to the formal semantics.

A method for representing and verifying assertion graphs symbolically is disclosed that provides an effective alternative for verifying families of properties. A symbolic indexing function provides a way of identifying assignments to Boolean variables with particular scalar cases. Formally defining a class of lattice domains based on symbolic indexing functions, provides an efficient symbolic manipulation technique using binary decision diagrams (BDDs).

Claim 8 sets forth a computer software product including one or more recordable media having executable instructions stored thereon which, when executed by a processing device, causes the processing device to perform, at least in part, a formal verification of a circuit or other finite-state system[1], said executable instructions causing the processing device to initialize a symbolic simulation relation[2] for an assertion graph[3] on a first

---

[1] "Intuitively, a model of a circuit or other finite state system can be simulated and the behavior of the model can be verified against properties expressed in an assertion graph language. Formal semantics of the assertion graph language explain how to determine if the model satisfies the property or properties expressed by the assertion graph. Two important characteristics of this type of verification system are the expressiveness of the assertion graph language and the computational efficiency of carrying out the verification." **(p. 7, lines 12-18).**
"For one embodiment, a finite state system can be formally defined on a nonempty finite set of states, S, as a nonempty transition relation, M, where (s1, s2) is an element of the transition relation, M, if there exists a transition in the finite state system from state s1 to state s2 and both s1 and s2 are elements of S. M is called a model of the finite state system." **(p. 7, lines 19-23).**
[2] "For one possible embodiment, an assertion graph, G, can be defined on a finite nonempty set of vertices, V, to include an initial vertex, vI; a set of edges, E, having one or more copies of outgoing edges originating from each vertex in V; a label mapping, Ant, which labels an edge, e, with an antecedent Ant(e); and a label mapping, Cons, which labels an edge, e, with a consequence, Cons(e). When an outgoing edge, e, originates from a vertex, v, and terminates at vertex, v', the original vertex, v, is called the head of e (written v = Head(e))" **(p. 9, lines 11-18).** "define a simulation relation sequence, Sim_k: $E \rightarrow P(S)$, mapping edges between vertices in G into state subsets in [a model] M as follows:
$Sim_1(e) = Ant(e)$ if $Head(e)=vI$, otherwise
$Sim_1(e) = \{ \}$;..." **(p.16, line 24 through p. 17, line 2).**
"Box 311 represents initially assigning an empty set to the simulation relation for all edges e in the assertion graph that do not begin at initial vertex vI, and initially assigning Ant(e) to the simulation relation

symbolic lattice domain[4]; and compute the symbolic simulation relation[5] for the assertion graph on the first symbolic lattice domain to verify the assertion graph[6] according to a normal satisfiability criteria[7].

---

for all edges e that do begin at initial vertex v1." **(p. 17, lines 14-17, Fig. 3a, 311)** "In block 611, the antecedent sets are strengthened for each edge in the assertion graph." **(p. 21, lines 8-9, Fig. 6a, 611)** "In block 621, the strengthened antecedent set fixpoint for each edge e (denoted Ant*(e)) in assertion graph G is computed." **(p. 21, lines 15-16, Fig. 6b, 621)**

[3] "For one embodiment, Figure 1b depicts an assertion graph, 102. The two types of labels used in the assertion graph have the following purposes: an antecedent represents a set of possible pre-existing states and stimuli to a circuit or finite state system to affect its behavior; a consequence represents a set of possible resulting states or behaviors to be checked through simulation of the circuit or finite state system. Antecedent and consequence labels are written as ai/ci for the edges of assertion graph 102." **(p. 9, lines 19-25, Fig. 1b)** "The abstracted assertion graph G_A is an assertion graph on a lattice domain (P_A, $\sqsubseteq_A$) having the same vertices and edges as G and for the abstracted antecedent labeling Ant_A and the abstracted consequence labeling $Cons_A$, $Ant_A(e)$=A(Ant(e)) and $Cons_A(e)$=A(Cons(e)) for all edges e in the assertion graphs G_A and G." **(p. 24, lines 14-18)**

[4] "It will be appreciated that the Union operation and the Intersect operation may also be interpreted as the Join operation and the Meet operation respectively." **(p. 17, lines 10-13)** "One lattice domain of interest is the set of all subsets of [a finite set of states] S, P(S) along with a subset containment relation, $\sqsubseteq$. The subset containment relation defines a partial order between elements of P(S), with the empty set as a lower bound and S as an upper bound. The set P(S) together with the subset containment relation, $\sqsubseteq$, are called a partially ordered system." **(p. 22, lines 7-11)** "For one embodiment an abstraction of the lattice domain (P(S), $\sqsubseteq$) onto a lattice domain (P, $\sqsubseteq_A$) can be defined by an abstraction function A mapping P(S) onto P… Figure 7 illustrates one embodiment of an abstraction function A." **(see p. 22, line 23 through p. 23, line 11; Fig. 7)**

[5] "$Sim_n(e)$ = Union ($Sim_{n-1}(e)$, ($Union_{\text{for all } e' \text{ such that } Tail(e')=Head(e)}$ (Intersect (Ant(e), Post($Sim_{n-1}(e')$))))),
for all n>1.

In the simulation relation defined above, the nth simulation relation in the sequence is the result of inspecting every state sequence along every I-path of lengths up to n. For any n>1, a state s is in the nth simulation relation of an edge e if it is either in the n-1th simulation relation of e, or one of the states in its pre-image set is in the n-1th simulation relation of an incoming edge e', and state s is in the antecedent set of e." **(p. 17, lines 3-10)** "For one embodiment, Figure 3a illustrates a method for computing the simulation relation for a model and an assertion graph." **(see p. 17, line 13 through p. 18, line 12; Fig. 3a, 312-317; Fig. 4)** "In block 612, a fixpoint simulation relation is computed using the antecedent strengthened assertion graph." **(p. 21, lines 9-11, Fig. 6a, 612)** "In block 622, a fixpoint simulation relation set for each edge e (denoted Sim*(e)) is computed using the strengthened antecedents computed for each edge in block 621." **(p. 21, lines 16-19, Fig. 6b, 622)**

[6] "The assertion graph can be seen as a monitor of the circuit, which can change over time. The circuit is simulated and results of the simulation are verified against consequences in the assertion graph. The antecedent sequence on a path selects which traces to verify against the consequences." **(p. 16, lines18-21)** "Comparing the final simulation relation for each edge, with the consequence set for that edge, indicates whether the model 101 strongly satisfies the assertion graph 201." **(see p. 18, lines 13-21, Fig. 4)** "Finally in block 613, the simulation relation sets are compared to the consequence sets to see if, for each edge, the simulation relation set is a subset of the consequence set, which is the necessary condition for satisfiability." **(p. 21, lines 11-13; Fig. 6a, 613)** "In block 623, the comparison is performed." **(p. 21, line 19, Fig. 6b, 623)** "For one embodiment, Figure 8b illustrates a method for implicit normal satisfiability using an abstracted simulation relation." **(see p. 26, line 7 through p. 27, line 8; Figs. 8b and 8c)**

[7] "Strong satisfiability, however, is inadequate for expressing justification properties, which are causes of effects, rather than effects of causes." **(p. 14, lines 4-6)** "For one embodiment, a normal semantics for assertion graphs that provides for justification properties may be formally defined." **(p. 15, lines 4-5)** "To

Claim 4 sets forth a computer software product including one or more recordable media having executable instructions stored thereon which, when executed by a processing device, causes the processing device to perform, at least in part, a formal verification of a circuit or other finite-state system[1], said executable instructions causing the processing device to initialize a symbolic simulation relation [2,8] for an assertion graph[9] on a first symbolic lattice domain[4,10], wherein the assertion graph on the first symbolic lattice

---

say that a state, s, satisfies an edge, e (denoted by s |= e), means that for every trace, t, starting from s and every path, p, starting from e, trace, t, satisfies path, p, under the consequence edge labeling, Cons, whenever trace, t, satisfies path, p, under the antecedent edge labeling, Ant. To say that the model M satisfies assertion graph G (denoted by M |= G), means that for any edge e beginning at initial vertex v1 in G, all states, s, in M satisfy edge e." **(p. 15, lines 10-16)** "In order to indicate normal satisfiability, a method is needed to propagate future antecedents backwards. For one embodiment, a method can be defined to strengthen the antecedent set of an edge e by intersecting it with the pre-image sets of antecedents on future edges." **(p. 18, lines 22-25)** "For one embodiment, Figure 3b illustrates a method for computing the strengthened antecedents for an assertion graph." **(see p. 19, lines 13 through p. 20, line 12; Figs. 3b and 5a)** "Figure 5b shows the final simulation relation resulting from iterations of the method of Figure 3a performed on the antecedent strengthened assertion graph 502 and using model 101. Comparing the final simulation relation labels for each edge, with the consequence set for that edge (as shown in assertion graph 202) indicates whether the model 101 strongly satisfies the strengthened assertion graph 502… but more importantly model 101 satisfies assertion graph 202 according to normal satisfiability as previously defined." **(p. 20, lines 13-26; Fig. 5b)** "For one embodiment, Figure 6a shows a method for computing the normal satisfiability of an assertion graph by a model." **(p. 21, lines 7-8; Fig. 6a)** "For one embodiment, Figure 6b illustrates, in finer detail, a method of computing normal satisfiability." **(p. 21, lines 14-15, Fig. 6b)** "Similarly, if methods herein previously disclosed determine that an abstracted model M_A satisfies a true abstraction G_A, then the original model M satisfies the original assertion graph G, according to the normal satisfiability criteria." **(p. 25, lines 1-4)**

[8] "For an assertion graph G and a model M=(Pre, Post), define an antecedent strengthening sequence, Ant_n: $E \rightarrow P(S)$, mapping edges between vertices in G into state subsets in M as follows.

$Ant_1(e) = Ant(e)$, and

$Ant_n(e) =$ Intersect ($Ant_{n-1}(e)$, ($\text{Union}_{\text{for all } e' \text{ such that Head}(e') = \text{Tail}(e)}$ $Pre(Ant_{n-1}(e'))$)), for all n>1.

In the antecedent strengthening sequence defined above, a state s is in the nth antecedent set of an edge e if it is a state in the n-1th antecedent set of e, and one of the states in a pre-image set of the n-1th antecedent set of an outgoing edge e'." **(p. 19, lines 5-14)** "For one embodiment, Figure 3b illustrates a method for computing the strengthened antecedents for an assertion graph." **(see p. 19, line 17 through p. 20, line 4; Fig. 3b)**

[9] "As an example of a justification property, one might wish to assert the following: if the system enters state s1, and does not start in state s1, then at the time prior to entering state s1, the system must have been in state s0. For one embodiment, Figure 2b depicts an assertion graph 202, which attempts to capture the justification property asserted in the above example." **(p. 14, lines 6-11; Fig. 2b)** "The abstracted assertion graph G_A is an assertion graph on a lattice domain (P_A, $\sqsubseteq_A$) having the same vertices and edges as G and for the abstracted antecedent labeling Ant_A and the abstracted consequence labeling $Cons_A$, $Ant_A(e)=A(Ant(e))$ and $Cons_A(e)=A(Cons(e))$ for all edges e in the assertion graphs G_A and G." **(p. 24, lines 14-18)**

[10] "Again, it will be appreciated that the Union operation and the Intersect operation may also be interpreted as the Join operation and the Meet operation respectively." **(p. 19, lines 14-16)**

domain is configurable to express a justification property [7,11] to verify by computing the symbolic simulation relation[5,6,12].

Claim 14 sets forth a computer implemented method for performing, at least in part, a formal verification of a circuit or other finite-state system[1], said method comprising: initializing a symbolic simulation relation[2,8] for an assertion graph[9] on a first symbolic lattice domain[4,10], wherein the assertion graph on the first symbolic lattice domain is configurable to express a justification property[7,11] to verify through computing the symbolic simulation relation[5,6,12].

Claim 16 sets forth a computer implemented method for performing, at least in part, a formal verification of a circuit or other finite-state system, said method comprising specifying a justification property with an assertion graph[2,7,8,9,11].

Claim 28 sets forth a computer implemented verification system for performing, at least in part, a formal verification of a circuit or other finite-state system[1], said system comprising: means for initializing a symbolic simulation relation[2,8] for an assertion graph[9,13] on a first symbolic lattice domain[4,10,14], wherein the assertion graph on the first

[11] "For example, Figure 5a shows iterations of antecedent strengthening of graph 202 on model 101." **(see p. 20, lines 4-12; Fig. 2, 202 and Fig. 5b, 502)**

[12] "Figure 5b shows the final simulation relation resulting from iterations of the method of Figure 3a performed on the antecedent strengthened assertion graph 502 and using model 101." **(see p. 20, lines 13-26; Fig. 5b)** "In block 622, a fixpoint simulation relation set for each edge e (denoted Sim*(e)) is computed using the strengthened antecedents computed for each edge in block 621." **(see p. 21, line 7 through p. 22, line 2; Figs. 6a and 6b)**

[13] "Formally defining a class of lattice domains based on symbolic indexing functions, provides an efficient symbolic manipulation technique using BDDs. Therefore previously disclosed methods for antecedent strengthening, abstraction, computing simulation relations, verifying satisfiability and implicit satisfiability may be extended to assertion graphs that are symbolically represented." **(p. 27, line 26 through p. 28, line 5)** "For one embodiment, an assertion graph G_S on a symbolic lattice domain $(\{B^m \to P\}, \sqsubseteq_S)$ can be set forth as a mapping $G_S(\underline{b})$ of m-ary boolean values $\underline{b}$ in B^m to scalar instances of assertion graph G_S on the original lattice domain $(P, \sqsubseteq)$ such that for the symbolic antecedent labeling Ant_S and the symbolic consequence labeling $Cons_S$,

$Ant_S(\underline{b})(e) = Ant_S(e)(\underline{b})$, and

$Cons_S(\underline{b})(e) = Cons_S(e)(\underline{b})$,

for all edges e in the assertion graph G_S. Figure 11a shows two assertion graphs, 1101 and 1102, on a lattice domain $(P, \sqsubseteq)$ and an assertion graph 1103 on the unary symbolic lattice domain 901 that symbolically encodes assertion graphs 1101 and 1102." **(see p. 29, line 11 through p. 30, line 19; Fig. 11a)**

[14] "For one embodiment, an m-ary symbolic extension of a lattice domain $(P, \sqsubseteq)$ can be set forth as a set of symbolic indexing functions $\{B^m \to P\}$ where B^m is the m-ary Boolean product." **(p. 28, lines 6-8)** "As an

symbolic lattice domain is configurable to express a justification property[7,11] to verify through computing the symbolic simulation relation[5,6,12,15]; means for computing the symbolic simulation relation[5,12,15] for the assertion graph[9,13] on the first symbolic lattice domain[4,10,14]; and means for checking the symbolic simulation relation[6] to verify a plurality of properties expressed by a plurality of corresponding assertion graph instances, having at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

Claim 5 sets forth the computer software product recited in Claim 4 which, when executed by a processing device, further causes the processing device to compute the symbolic simulation relation[5,12,15] for the assertion graph[9,13] on the first symbolic lattice domain[4,10,14]; and check the symbolic simulation relation[6] to verify a plurality of properties expressed by a plurality of assertion graph instances, having at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

Claim 15 sets forth the method recited in Claim 14 further comprising computing the symbolic simulation relation[5,12,15] for the assertion graph[9,13] on the first symbolic lattice domain[4,10,14]; and checking the symbolic simulation relation[6] to verify a plurality of properties expressed by a plurality of corresponding assertion graph instances, having at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

Claim 18 sets forth The method recited in Claim 17 further comprising computing a symbolic simulation relation[5,12,15] for the assertion graph[9,13] on the first symbolic lattice domain[4,10,14]; and checking the symbolic simulation relation with a symbolic consequence labeling for the assertion graph[6] on the first symbolic lattice domain according to a normal satisfiability criteria[7].

---

example of a symbolic lattice domain. Figure 9 depicts part of a unary symbolic lattice domain." **(see p. 28, line 24 through p. 29, line 4; Fig. 9)**

[15] "For one embodiment, Figure 12a illustrates a method for computing the simulation relation for a model and an assertion graph on the symbolic lattice domain ($\{B^m \to P\}, \sqsubseteq_S$)." **(see p. 30, line 20 through p. 33, line 12; Figs. 11b and 12a)**

VI. Grounds of Rejection to be Reviewed on Appeal

A. Claims **4**-5, **8**, **14**-15, **16**-18, **28** and 31-40 stand rejected under 35 USC § 112 as allegedly being indefinite and omitting essential structural relationships of elements.

B. Claims **4**-5, **8**, **14**-15, **16**-18, **28** and 31-40 stand rejected under 35 USC § 101 as allegedly lacking patentable utility.

VII. Argument

A. 35 U.S.C. § 112 REJECTIONS

Claims **4**-5, **8**, **14**-15, **16**-18, **28** and 31-40 stands rejected under 35 USC § 112, second paragraph, as allegedly being indefinite for failing to point out and distinctly claim the invention by omitting essential structural relationships of elements, the Office Action (p. 4, § 3.1) alleging that according to MPEP § 2172.01, the claims must necessarily provide a linkage between circuit verification and manipulation/initialization of [a symbolic simulation relation for] an assertion graph in a first symbolic lattice domain and further, that they fail to do so.

The Office Action (p. 4, § 3.2) also states that it is not clear what the metes and bounds are of the term "other finite state system," which is set forth in independent claims 4, 8, 14, 16 and 28. The examiner alleges that the specification fails to provide a definition of the instant term at issue.

<u>1. Claims 4, 8, 14, 16 and 28 Are Not Indefinite.</u>

The issue of definiteness is whether, in light of the teachings of the prior art and of the particular invention, the claims set out and circumscribe a particular area with a reasonable degree of precision and particularity. *In re Moore*, 439 F.2d 1232, 1235, 169 USPQ 236, 238 (CCPA 1971).

Claim 4, for example, sets forth (emphasis added):

> 4. (Previously Presented) A computer software product including one or more recordable media having executable instructions stored thereon which, when executed by a processing device, causes the processing device to perform, at least in part, a formal <u>verification of a circuit or other finite-state system</u>, said executable instructions causing the processing device to:
>
> initialize a symbolic simulation relation for an assertion graph on a first symbolic lattice domain, wherein <u>the assertion graph</u> on the first symbolic lattice domain <u>is configurable to express a justification property to verify by computing the symbolic simulation relation</u>.

With regard to whether the instant claim provides linkage between circuit verification and manipulation/initialization of a symbolic simulation relation for an assertion graph in a first symbolic lattice domain, appellant respectfully submits that claim 1 sets forth that the assertion graph on the first symbolic lattice domain is configurable to express a justification property to verify by computing the symbolic simulation relation. Thus the instant claim clearly does provide linkage between circuit verification and manipulation/initialization of a symbolic simulation relation for an assertion graph in a first symbolic lattice domain.

The test for definiteness under 35 U.S.C. § 112 is whether those skilled in the art would understand what is claimed when the claim is read in light of the specification. *Orthokinetics, Inc. v. Safety Travel Chairs, Inc.*, 806 F.2d 1565, 1576, 1 USPQ2d, 1081, 1088 (Fed. Cir. 1986).

Appellant respectfully submits that the specification has set forth a full and clear description of the claimed subject matter. For example, with regard to initializing a symbolic simulation relation, the specification discloses (p. 9, lines 11-18) that:

> For one possible embodiment, an assertion graph, G, can be defined on a finite nonempty set of vertices, V, to include an initial vertex, vI; a set of edges, E, having one or more copies of outgoing edges originating from each vertex in V; a label mapping, Ant, which labels an edge, e, with an antecedent Ant(e); and a label mapping, Cons, which labels an edge, e, with a consequence, Cons(e). When an outgoing edge, e, originates from a vertex, v, and terminates at vertex, v', the original vertex, v, is called the head of e (written v = Head(e)).

and further discloses (p.16, line 22 through p. 17, line 2) that:

> For one embodiment, a simulation relation sequence can be defined for model checking according to the strong satisfiability criteria defined above. For an assertion graph G and a model M=(Pre, Post), define a simulation relation sequence, Sim_n: E→P(S), mapping edges between vertices in G into state subsets in M as follows:
>
> $\mathrm{Sim}_1(e)$ = Ant(e) if Head(e)=vI, otherwise
>
> $\mathrm{Sim}_1(e)$ = { };...

and further discloses (p. 17, lines 14-17; Fig. 3a, 311) that:

> Box 311 represents initially assigning an empty set to the simulation relation for all edges e in the assertion graph that do not begin at initial vertex vI, and initially assigning Ant(e) to the simulation relation for all edges e that do begin at initial vertex vI.

It will be appreciated that there is a direct correspondence between the formal definition of $\mathrm{Sim}_1(e)$ and the initialization performed by Box 311 of Figure 3a. Appellant respectfully submits that at least in light of the above disclosure set forth by the specification, the claims set out and circumscribe initializing a symbolic simulation relation for symbolic model checking with a reasonable degree of precision and particularity. The specification further discloses (p. 19, lines 5-14) that:

> For an assertion graph G and a model M=(Pre, Post), define an antecedent strengthening sequence, Ant_n, E→P(S), mapping edges between vertices in G into state subsets in M as follows:
>
> $\mathrm{Ant}_1(e)$ = Ant(e), and
>
> $\mathrm{Ant}_n(e)$ = Intersect ($\mathrm{Ant}_{n-1}(e)$, ($\mathrm{Union}_{\text{for all } e' \text{ such that Head}(e')=\text{Tail}(e)}$ Pre($\mathrm{Ant}_{n-1}(e')$))), for all n>1.
>
> In the antecedent strengthening sequence defined above, a state s is in the nth antecedent set of an edge e if it is a state in the n-1th antecedent set of e, and one of the states in a pre-image set of the n-1th antecedent set of an outgoing edge e'.

and further discloses (see p. 19, line 17 through p. 20, line 4; Fig. 3b) that:

For one embodiment, Figure 3b illustrates a method for computing the strengthened antecedents for an assertion graph.

and further discloses (p. 21, lines 15-16; Fig 6b, 621) that:

In block 621, the strengthened antecedent set fixpoint for each edge e (denoted Ant*(e)) in assertion graph G is computed.

Appellant respectfully submits that at least in light of the above disclosure set forth by the specification, the claims set out and circumscribe initializing a symbolic simulation relation for an assertion graph configurable to express a justification property with a reasonable degree of precision and particularity.

With regard to initializing a symbolic simulation relation for an m-ary symbolic extension of a lattice domain, the specification further discloses (p. 30, line 20 through p. 31 line 1) that:

Given a model M_S on the symbolic lattice domain $(\{B^m \rightarrow P\}, \sqsubseteq_S)$, and an assertion graph G_S on the symbolic lattice domain $(\{B^m \rightarrow P\}, \sqsubseteq_S)$ having edges $(\underline{v}, \underline{v}')$ and $('\underline{v}, \underline{v})$ where $\underline{v}'$ denotes the successors of $\underline{v}$, and $'\underline{v}$ denotes the predecessors of $\underline{v}$, a method to symbolically compute the simulation relation sequence of G_S can be formally defined. For one embodiment, a symbolic simulation relation sequence $\mathrm{Sim}_S(\underline{v}, \underline{v}')$ can be defined for model checking according to the strong satisfiability criteria as follows:

$\mathrm{Sim}_{S1}(\underline{v}, \underline{v}') = (\mathrm{initE}(\underline{v}, \underline{v}') \text{ AND } U) \text{ Meet}_S \text{ Ant}_S(\underline{v}, \underline{v}')$

where initE is a Boolean predicate for the set of edges outgoing from vI.

and further discloses (p. 31, lines 10-15; Fig. 12a, 1211) that:

Box 1211 represents initially assigning

$Z = (\mathrm{initE}(\underline{v}, \underline{v}') \wedge U) \cap_S \mathrm{Ant}_S(\underline{v}, \underline{v}')$

to the simulation relation for all edges $(\underline{v}, \underline{v}')$ in the assertion graph that do not begin at initial vertex vI, and initially assigning

$\mathrm{Ant}_S(\underline{v}, \underline{v}') = (\mathrm{initE}(\underline{v}, \underline{v}') \wedge U) \cap_S \mathrm{Ant}_S(\underline{v}, \underline{v}')$

to the simulation relation for all edges $(\underline{v}, \underline{v}')$ that do begin at initial vertex vI.

It will also be appreciated that there is a direct correspondence between the formal definition of $\mathrm{Sim}_{S1}(\underline{v}, \underline{v}')$ and the initialization performed by Box 1211 of Figure 12a. Appellant respectfully submits that at least in light of the above disclosure set forth by the specification, the claims set out and circumscribe initializing a symbolic simulation relation for an assertion graph on a first symbolic lattice domain with a reasonable degree of precision and particularity. The specification further discloses (p. 33, lines 12-17) that:

> For one embodiment, an antecedent strengthening sequence $\text{Ant}_S(\underline{v}', \underline{v})$ can be defined for model checking according to the normal satisfiability criteria as follows:
> $\text{Ant}_{S1}(\underline{v}', \underline{v}) = \text{Ant}_S(\underline{v}', \underline{v})$, and
> $\text{Ant}_{Sn}(\underline{v}', \underline{v}) = \text{Meet}_S\ (\text{Ant}_{Sn-1}(\underline{v}', \underline{v}), (\text{Join}_{S\text{ for all b in Bm}}\ \text{Pre}_S(\text{Sim}_{Sn-1}(\underline{v}, \underline{v}'))[b/\underline{v}']\ ))$, for all $n>1$.

and further discloses (see p. 33, line 18 through p. 34, line 9; Fig. 12b) that:

> For one embodiment, Figure 12b illustrates a method for computing the strengthened antecedents for an assertion graph on a symbolic lattice domain.

Appellant respectfully submits that at least in light of the above disclosure set forth by the specification, the claims set out and circumscribe, with a reasonable degree of precision and particularity, initializing a symbolic simulation relation for an assertion graph on a first symbolic lattice domain, wherein the assertion graph on the first symbolic lattice domain is configurable to express a justification property.

The specification also discloses that (p. 16, lines 18-21) that:

> The assertion graph can be seen as a monitor of the circuit, which can change over time. The circuit is simulated and results of the simulation are verified against consequences in the assertion graph. The antecedent sequence on a path selects which traces to verify against the consequences.

The specification further discloses (p. 17, lines 3-10) that:

> $\text{Sim}_n(e) = \text{Union}\ (\text{Sim}_{n-1}(e), (\text{Union}_{\text{for all e' such that Tail(e')=Head(e)}}\ (\text{Intersect}\ (\text{Ant}(e), \text{Post}(\text{Sim}_{n-1}(e')))\ )))$, for all $n>1$.
> In the simulation relation defined above, the nth simulation relation in the sequence is the result of inspecting every state sequence along every I-path of lengths up to n. For any $n>1$, a state s is in the nth simulation relation of an edge e if it is either in the n-1th simulation relation of e, or one of the states in its pre-image set is in the n-1th simulation relation of an incoming edge e', and state s is in the antecedent set of e. It will be appreciated that the Union operation and the Intersect operation may also be interpreted as the Join operation and the Meet operation respectively.

and further discloses (see p. 17, lines 13-14, Fig. 3a, 312-317) that:

> For one embodiment, Figure 3a illustrates a method for computing the simulation relation for a model and an assertion graph.

and further discloses (p. 21, lines 9-11; Fig. 6a, 612) that:

> In block 612, a fixpoint simulation relation is computed using the antecedent strengthened assertion graph.

and further discloses (p. 21, lines 16-19; Fig. 6b, 622) that:

> In block 622, a fixpoint simulation relation set for each edge e (denoted Sim*(e)) is computed using the strengthened antecedents computed for each edge in block 621.

Appellant respectfully submits that at least in light of the above disclosure set forth by the specification, the claims set out and circumscribe, with a reasonable degree of precision and particularity, initializing a symbolic simulation relation for an assertion graph on a first symbolic lattice domain, wherein the assertion graph on the first symbolic lattice domain is configurable to express a justification property to verify by computing the symbolic simulation relation

Therefore appellant respectfully submits that when the claim is read in light of the specification, one skilled in the art would understand what is claimed including the relation between circuit verification and manipulation/initialization of a symbolic simulation relation for an assertion graph in a first symbolic lattice domain.

With regard to whether the specification provides a definition of the instant term "other finite state system," appellant respectfully submits that the specification discloses that (p. 7, lines 12-18; emphasis added) that:

> Intuitively, a model of <u>a circuit or other finite state system can be simulated and the behavior</u> of the model <u>can be verified against properties expressed in an assertion graph language</u>. Formal semantics of the assertion graph language explain how to determine if the model satisfies the property or properties expressed by the assertion graph. Two important characteristics of this type of verification system are the expressiveness of the assertion graph language and the computational efficiency of carrying out the verification.

and further discloses (p. 7, lines 19-23; emphasis added) that:

> For one embodiment, <u>a finite state system can be formally defined</u> on a nonempty finite set of states, S, <u>as a nonempty transition relation, M</u>, where (s1, s2) is an element of the transition relation, M, if there exists a transition in the finite state system from state s1 to state s2 and both s1 and s2 are elements of S. <u>M is called a model of the finite state system</u>.

Thus the specification clearly does provide a definition of the instant term, such that when the claim is read in light of the specification, one skilled in the art would understand what is claimed including the term "other finite state system."

Therefore, appellant respectfully submits that in light of the specification, those skilled in the art would understand what is claimed. Accordingly, appellant submits that claims 4, 8, 14, 16 and 28 sufficiently set out and circumscribe respectively, their particular areas with a reasonable degree of precision and particularity required by 35 USC § 112, second paragraph.

2. Claims 5, 15 and 18 are Not Indefinite.

Claim 5, for example, sets forth:

> 5. (Original) The computer software product recited in Claim 4 which, when executed by a processing device, further causes the processing device to:
>
> compute the symbolic simulation relation for the assertion graph on the first symbolic lattice domain; and
>
> check the symbolic simulation relation to verify a plurality of properties expressed by a plurality of assertion graph instances, having at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

With regard to computing the symbolic simulation relation, the specification discloses (p.16, line 22 through p. 17, line 12) that:

> For one embodiment, a simulation relation sequence can be defined for model checking according to the strong satisfiability criteria defined above. For an assertion graph G and a model M=(Pre, Post), define a simulation relation sequence, Sim_n: E→P(S), mapping edges between vertices in G into state subsets in M as follows:
>
> $\mathrm{Sim}_1(e)$ = Ant(e) if Head(e)=v1, otherwise
>
> $\mathrm{Sim}_1(e)$ = { };
>
> $\mathrm{Sim}_n(e)$ = Union ($\mathrm{Sim}_{n-1}(e)$,
>
> ($\mathrm{Union}_{\text{for all e' such that Tail(e')=Head(e)}}$ (Intersect (Ant(e), Post(Sim_{n-1}(e')))))), for all n>1.
>
> In the simulation relation defined above, the nth simulation relation in the sequence is the result of inspecting every state sequence along every I-path of lengths up to n. For any n>1, a state s is in the nth simulation relation of an edge e if it is either in the n-1th simulation relation of e, or one of the states in its pre-image set is in the n-1th simulation relation of an incoming edge e', and state s is in the antecedent set of e. It will be appreciated that the Union operation and the Intersect operation may also be interpreted as the Join operation and the Meet operation respectively.

and further discloses (p. 17, line17 through p. 18, line 2; Fig. 3a, 312-317) that:

> Box 312 represents marking all edges in the assertion graph active. Box 313 represents testing the assertion graph to identify any active edges. If no active edges are identified, then the method is complete. Otherwise, an active edge, e, is selected and marked not active as represented by box 314. Box 315 represents recomputing the simulation relation for edge, e, by adding to the simulation relation for edge e, any states which are in both the antecedent set for edge e and the post-image set for the simulation relation of any incoming edge, e', to e. Box 316 represents testing the simulation relation for edge e to determine if it was changed by the recomputation. If it has changed, all outgoing edges from e are marked as active, as represented by Box 317. In any case, the method flow returns to the test for active edges represented by Box 313.

It will be appreciated that there is a direct correspondence between the formal definition of $\mathrm{Sim}_n(e)$ and the iterative computing performed by Box 315 of Figure 3a. The specification further discloses (p. 20, lines13-15; Fig. 5b) that:

> Figure 5b shows the final simulation relation resulting from iterations of the method of Figure 3a

performed on the antecedent strengthened assertion graph 502 and using model 101.

Appellant respectfully submits that at least in light of the above disclosure set forth by the specification, the claims set out and circumscribe computing a symbolic simulation relation for symbolic model checking with a reasonable degree of precision and particularity.

With regard to computing a symbolic simulation relation for an m-ary symbolic extension of a lattice domain, the specification further discloses (p. 30, line 20 through p. 31 line 7) that:

> Given a model M_S on the symbolic lattice domain $(\{B^m \to P\}, \sqsubseteq_S)$, and an assertion graph G_S on the symbolic lattice domain $(\{B^m \to P\}, \sqsubseteq_S)$ having edges $(\underline{v}, \underline{v}')$ and $(\underline{v}', \underline{v})$ where $\underline{v}'$ denotes the successors of $\underline{v}$, and $\underline{v}'$ denotes the predecessors of $\underline{v}$, a method to symbolically compute the simulation relation sequence of G_S can be formally defined. For one embodiment, a symbolic simulation relation sequence $Sim_S(\underline{v}, \underline{v}')$ can be defined for model checking according to the strong satisfiability criteria as follows:
>
> $Sim_{S0}(\underline{v}, \underline{v}') = (initE(\underline{v}, \underline{v}')$ AND U) $Meet_S\ Ant_S(\underline{v}, \underline{v}')$
>
> where initE is a Boolean predicate for the set of edges outgoing from vI, , and
>
> $Sim_{Sn}(\underline{v}, \underline{v}') = Join_S\ (Sim_{Sn-1}(\underline{v}, \underline{v}'), (Join_{S\ \text{for all}\ \underline{b}\ \text{in}\ B^m}$ (
>
> $Meet_S\ (Ant(\underline{v}, \underline{v}'), Post_S(Sim_{Sn-1}(\underline{v}'', \underline{v}))))[\underline{b}/\underline{v}'']))$, for all n>1
>
> where $Join_S$ and $Meet_S$ are the join, $\sqcup_S$, and meet, $\sqcap_S$, operators for the symbolic lattice domain $(\{B^m \to P\}, \sqsubseteq_S)$ and $[\underline{b}/\underline{v}'']$ denotes replacing each occurrence of $\underline{v}''$ in the previous expression with $\underline{b}$.

and further discloses (p. 31, lines 16-24; Fig. 12a, 1215-1216) that:

> Box 1215 represents recomputing the simulation relation for edge $(\underline{v}, \underline{v}')$ by adding to the simulation relation for edges $(\underline{v}, \underline{v}')$, any states which are in both the antecedent set for edges $(\underline{v}, \underline{v}')$ and the post-image set for the simulation relation of any incoming edges $(\underline{v}'', \underline{v})$ to $(\underline{v}, \underline{v}')$ produced by substituting any $\underline{b}$ in B^m for $\underline{v}''$. Box 1216 represents testing the simulation relation labeling for edges $(\underline{v}, \underline{v}')$ to determine if it was changed by the recomputation. If it has changed, the method flow returns to the recomputation of simulation relation for edges $(\underline{v}, \underline{v}')$, represented by Box 1215. Otherwise a fixpoint has been reached and the method terminates at box 1216.

It will also be appreciated that there is a direct correspondence between the formal definition of $Sim_{Sn}(\underline{v}, \underline{v}')$ and the iterated computing performed by Box 1215 of Figure 12a. Appellant respectfully submits that at least in light of the above disclosure set forth by the specification, the claims set out and circumscribe computing the symbolic simulation relation for the assertion graph on the first symbolic lattice domain with a reasonable degree of precision and particularity.

As relied upon above with regard to claims 4, 8, 14, 16 and 28, the test for definiteness under 35 U.S.C. § 112 is whether those skilled in the art would understand what is claimed when the claim is read in light of the specification. *Orthokinetics, Inc., supra.*

Appellant respectfully submits that in light of the specification, those skilled in the art would understand what is claimed, including the relation between circuit verification and the claimed computing of the symbolic simulation relation for the assertion graph on the first symbolic lattice domain and checking of the symbolic simulation relation to verify a plurality of properties as set forth by claims 5, 15 and 18.

B. 35 U.S.C. § 101 REJECTIONS

Claims **4**-5, **8**, **14**-15, **16**-18, **28** and 31-40 stand rejected under 35 USC § 101 as allegedly lacking patentable utility, the Office Action (p. 2, § 2.1) alleging that the claims fail to disclose a tangible, concrete and useful result. The Office Action states that although the specification discloses automated design verification for large scale integrated circuits, the claim language includes the term "other finite state system," as set forth in independent claims 4, 8, 14, 16 and 28. The examiner alleges that the specification fails to provide a definition of the instant term at issue and that the instant term is not tied to the actual physical world.

1. Claims 4, 8, 14, 16 and 28 Are Directed to Patentable Subject Matter.

35 USC § 101 clearly states that, "Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title."

Claim 4, for example, sets forth:

> 4. (Previously Presented) A computer software product including one or more recordable media having executable instructions stored thereon which, when executed by a processing device, causes the processing device to perform, at least in part, a formal verification of a circuit or other finite-state system, said executable instructions causing the processing device to:
>
> initialize a symbolic simulation relation for an assertion graph on a first symbolic lattice domain, wherein the assertion graph on the first symbolic lattice domain is configurable to express a justification property to verify by computing the symbolic simulation relation.

and Claim 8 sets forth:

> 8. (Previously Presented) A computer software product including one or more recordable media having executable instructions stored thereon which, when executed by a processing device, causes the processing device to perform, at least in part, a formal verification of a circuit or other finite-state system, said executable instructions causing the processing device to:
>
> initialize a symbolic simulation relation for an assertion graph on a first symbolic lattice domain; and
>
> compute the symbolic simulation relation for the assertion graph on the first symbolic lattice domain to verify the assertion graph according to a normal satisfiability criteria.

The Office Action of December 7, 2006 (p. 3 § 2.2) states that the manipulation of [symbolic simulation relations for] an assertion graph and the subsequent verification of a symbolic simulation relation is merely teaching the manipulation of an abstract concept (citing MPEP § 2106.02, rev 5, 2006).

But, appellant respectfully notes that the cited section (§ 2106.02) refers to processes that simply manipulate abstract ideas without some claimed practical application, whereas all the instant claims set forth the practical application of performing formal verifications of circuits or other finite-state systems.

On the other hand, with regard to claims 4 and 8, appellant respectfully submits that MPEP § 2106.01, par. 2, states that:

> When functional descriptive material is recorded on some computer-readable medium, it becomes structurally and functionally interrelated to the medium and will be statutory in most cases since use of technology permits the function of the descriptive material to be realized.

Further the MPEP § 2106.01, par. 6, states that:

> ...a claimed computer-readable medium encoded with a computer program is a computer element which defines structural and functional interrelationships between the computer program and the rest of the computer which permit the computer program's functionality to be realized, and is thus statutory. See *Lowry*, 32 F.3d at 1583-84, 32 USPQ2d at 1035. Accordingly, it is important to distinguish claims that define descriptive material *per se* from claims that define statutory inventions.

Claim 14 also sets forth:

> 14. (Previously Presented) A computer implemented method for performing, at least in part, a formal verification of a circuit or other finite-state system, said method comprising:
>
> initializing a symbolic simulation relation for an assertion graph on a first symbolic lattice domain, wherein the assertion graph on the first symbolic lattice domain is configured to express a justification property to verify through computing the symbolic simulation relation.

With regard to the computer implemented method claims 14 and 16, appellant respectfully submits that MPEP § 2106.01, par. 8, states that:

> When a computer program is claimed in a process where the computer is executing the computer program's instructions, USPTO personnel should treat the claim as a process claim.

The Office Action (p. 2-3, § 2.1) further states that although the specification discloses automated design verification for large scale integrated circuits, the claims include the term "other finite state system," so the claim language, even in light of the specification, fails to disclose a useful result because the claimed invention could also be used to verify some abstract finite state system.

Appellant respectfully submits that the examiner is in error for reasoning that if the claimed invention could possibly be used to verify some abstract finite state system, then it can have no credible utility. Appellant submits that perhaps any patentable apparatus may have such a non-statutory use, which by itself, would not be sufficient to establish patentability (use as landfill, for example) but a mere possibility of the existence of such a non-statutory use is insufficient to negate an otherwise credible assertion of utility for the apparatus in establishing it as statutory subject matter.

An analysis of the instant claims must be performed in order to make a determination of whether the subject matter is statutory. Such analysis should not occur in a vacuum but should correlate each claim element with corresponding structures, materials or acts set forth in the specification.

Appellant respectfully submits that the specification discloses (p. 7, lines 12-18; emphasis added) that:

> Intuitively, a model of a circuit or other finite state system can be simulated and the behavior of the model can be verified against properties expressed in an assertion graph language. Formal semantics of the assertion graph language explain how to determine if the model satisfies the property or properties expressed by the assertion graph. Two important characteristics of this type of verification system are the expressiveness of the assertion graph language and the computational efficiency of carrying out the verification.

and further discloses (p. 7, lines 19-23; emphasis added) that:

> For one embodiment, a finite state system can be formally defined on a nonempty finite set of states, S, as a nonempty transition relation, M, where (s1, s2) is an element of the transition relation, M, if there exists a transition in the finite state system from state s1 to state s2 and both s1 and s2 are elements of S. M is called a model of the finite state system.

Thus the specification provides a definition of a finite state system, such that the claimed subject matter would be understood by a person of skill in the art of formal verification in the context of the entire patent.

The Federal Circuit makes it clear that the ordinary and customary meaning of a claim term is the meaning that the term would have to a person of ordinary skill in the art in question at the time of the invention. "The person of ordinary skill in the art is deemed to read the claim term not only in the context of the particular claim in which the disputed term appears, but in the context of the entire patent, including the specification." *Phillips v. AWH Corp.*, 415 F.3d at 1313.

For example, the specification further discloses (p. 1, lines 10-13) that:

> As hardware and software systems become more complex there is a growing need for automated formal verification methods. These methods are mathematically based techniques and languages that help detect and prevent design errors thereby avoiding losses in design effort and financial investment.

and further discloses (p. 1, line 17 through p. 2, line 5, emphasis added) that:

> There are <u>two well-established symbolic methods for automatically verifying</u> such <u>properties of circuits and finite state systems</u> that are currently <u>considered to be significant</u>. The two most significant prior art methods are known as classical Symbolic Model Checking (SMC) and Symbolic Trajectory Evaluation (STE).
>
> Classical SMC is more widely known and more widely received in the formal verification community. It involves building a finite model of a system as a set of states and state transitions and checking that a desired property holds in the model. An exhaustive search of all possible states of the model is performed in order to verify desired properties. The high level model can be expressed as temporal logic with the system having finite state transitions or as two automata that are compared according to some definition of equivalence. <u>A representative of classical SMC from Carnegie Mellon University known as SMV</u> (Symbolic Model Verifier) <u>has been used for verifying circuit designs and protocols</u>. Currently <u>these techniques are being applied also to software verification</u>.

and also discloses (p. 2, line 18-22) that:

> The second and less well-known technique, <u>STE, is a lattice based model checking technique</u>. It is more suitable <u>for verifying</u> properties of <u>systems with very large state spaces</u> (specifiable in thousands or tens of thousands of state encoding variables) because the number of variables required depends on the assertion being checked rather than on the system being verified.

Therefore the specification provides a specific, substantial, and credible assertion of a well-established utility such that a person of ordinary skill in the art would appreciate why the invention is useful based on the characteristics of the invention (e.g., the technique is suitable for automated verification of more generalized properties in complex circuitry, protocols and software systems [i.e. or other finite state systems] with very large state spaces).

Thus the claims, in the context of the entire patent, would not be understood by a person of skill in the art of formal verification to merely teach manipulation of an abstract concept, but rather to have a practical application in the technical arts.

To facilitate formal verification of circuits or other finite-state systems, these manipulations and/or computations of symbolic simulation relations for assertion graphs perform transformations and/or reductions of data structures representative of or constituting physical activity such as internal machine state transitions and/or objects such as circuits, networks or other finite state systems.

The necessary element is sometimes phrased in terms of requiring a transformation or reduction of 'subject matter.' In Schrader, the phrase 'subject matter' was determined not to be limited to tangible articles or objects, but also includes intangible subject matter, such as data or signals, representative of or constituting physical activity or objects. *Schrader*, 22 F.3d at 295, 30 USPQ2D (BNA) at 1459.

The Supreme Court held that the focus in any statutory subject matter analysis be on the claim as a whole, stating "When a claim containing a mathematical formula implements or applies that formula in a structure or process which, when considered as a whole, is performing a function which the patent laws were designed to protect (e.g., transforming or reducing an article to a different state or thing, then the claim satisfies the requirements of § 101." *In re Alappat*, 33 F.3d 1526 at 1543, 31 USPQ2d 1545 at 1556 (Fed. Cir. 1994) (quoting *Diehr*, 450 U.S. at 192, 209 USPQ at 10).

The Federal Circuit has pointed out (emphasis provided by the Fed. Cir.) that, "It is thus not necessary to determine whether a claim contains, as merely a part of the whole, any mathematical subject matter which standing alone would not be entitled to patent protection. Indeed, because the dispositive inquiry is whether the claim as a whole is directed to statutory subject matter, it is irrelevant that a claim may contain, as part of the whole, subject matter which would not be patentable by itself." A claim drawn to subject matter otherwise statutory does not become nonstatutory simply because it uses a mathematical formula, [mathematical equation, mathematical algorithm,] computer program or digital computer." *Id* at 1543-1544 (quoting *Diehr*, 450 U.S. at 187, 209 USPQ at 5).

Accordingly, for the reasons given above, appellant respectfully submits that claims 4, 8, 14, 16 and 28 have patentable utility and therefore, are directed to patentable subject matter.

2. Claim 28 Is Directed to Patentable Subject Matter

With regard to claim 28, of December 7, 2006 (p. 3 § 2.3) states that the recited means could be interpreted to be programmed methods and as such, amount to software only. The examiner then reasons that the claimed system is composed of functional descriptive material, and as such is non-statutory (citing MPEP § 2106.01, rev 5, 2006).

Appellant respectfully submits that the examiner is in error, for misinterpreting the cited section to suggest that all functional descriptive material is non-statutory. The relevant portion of § 2106.01, states that (emphasis added):

> Computer programs are often recited as part of a claim. USPTO personnel should determine whether the computer program is being claimed as part of an otherwise statutory manufacture or machine. In such a case, the claim remains statutory irrespective of the fact that a computer program is included in the claim. The same result occurs when a computer program is used in a computerized process where the computer executes the instructions set forth in the computer program. <u>Only when</u> the claimed <u>invention taken as a whole is directed to a mere program listing</u>, i.e., to only its description or expression, <u>is it descriptive material *per se*</u> and hence nonstatutory.

Appellant respectfully submits that claim 28 is not directed to a computer program *per se*, but rather to a computer implemented verification system having means which permit the computer programs' functionality to be realized.

Claim 28 sets forth:

> 28. (Previously Presented) A computer implemented verification system for performing, at least in part, a formal verification of a circuit or other finite-state system, said system comprising:
>
> means for initializing a symbolic simulation relation for an assertion graph on a first symbolic lattice domain, wherein the assertion graph on the first symbolic lattice domain is configured to express a justification property to verify through computing the symbolic simulation relation;
>
> means for computing the symbolic simulation relation for the assertion graph on the first symbolic lattice domain; and
>
> means for checking the symbolic simulation relation to verify a plurality of properties expressed by a plurality of corresponding assertion graph instances, having at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

An analysis of the instant claims must be performed in order to make a determination of whether the subject matter is statutory. Such analysis should not occur in a vacuum but should correlate each claim element with corresponding structures, materials or acts set forth in the specification.

Appellant respectfully submits that the specification discloses (p. 35, lines 4-7) that:

> For example, Figure 13 illustrates a computer system to perform computations, for one such embodiment. Computer system 1322 is connectable with various storage, transmission and I/O devices to receive data structures and programmed methods.

and further discloses (p. 35, lines 12-26) that:

> Components of either or both of the data structures and programmed methods may be stored or transmitted on devices such as removable storage disks 1325, which may be accessed through an access device 1326 in computer system 1322 or in a storage serving system 1321. Storage serving system 1321 or computer system 1322 may also include other removable storage devices or non-removable storage devices suitable for storing or transmitting data structures 1301 or programmed methods 1302. Component data structures and programmed methods may also be stored or transmitted on devices such as network 1324 for access by computer system 1322 or entered by users through I/O device 1323. It will be appreciated that systems such as the one illustrated are commonly available and widely used in the art of designing finite state hardware and software systems. It will also be appreciated that the complexity, capabilities, and physical forms of such design systems improves and changes rapidly, and therefore understood that the design system illustrated is by way of example and not limitation.

Thus the specification provides a description of a computer implemented verification system having means which permit the computer programs' functionality to be realized, such that the claimed subject matter would be understood by a person of skill in the art of formal verification in the context of the entire patent.

Accordingly, for the reasons given above, appellant respectfully submits that claim 28 is directed to patentable subject matter.

Conclusion

Appellant submits that all claims now pending are in condition for allowance. Such action is earnestly solicited at the earliest possible date. If there is a deficiency in fees, please charge our Deposit Acct. No. 50-0221.

Respectfully submitted,

Date: June 11, 2007

/s/Lawrence M. Mennemeier/
Lawrence M. Mennemeier
Reg. No. 51,003

INTEL CORPORATION
c/o INTELLEVATE LLP
P.O. Box 52050
Minneapolis, MN 55402
(408) 765-2194

VIII. <u>Claims Appendix: Claims Allowed and Involved in Appeal (Clean Copy)</u>

1-3. (Canceled)

4. (Previously Presented) A computer software product including one or more recordable media having executable instructions stored thereon which, when executed by a processing device, causes the processing device to perform, at least in part, a formal verification of a circuit or other finite-state system, said executable instructions causing the processing device to:

initialize a symbolic simulation relation for an assertion graph on a first symbolic lattice domain, wherein the assertion graph on the first symbolic lattice domain is configurable to express a justification property to verify by computing the symbolic simulation relation.

5. (Original) The computer software product recited in Claim 4 which, when executed by a processing device, further causes the processing device to:

compute the symbolic simulation relation for the assertion graph on the first symbolic lattice domain; and

check the symbolic simulation relation to verify a plurality of properties expressed by a plurality of assertion graph instances, having at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

6-7. (Canceled)

8. (Previously Presented) A computer software product including one or more recordable media having executable instructions stored thereon which, when executed by a processing device, causes the processing device to perform, at least in part, a formal verification of a circuit or other finite-state system, said executable instructions causing the processing device to:

initialize a symbolic simulation relation for an assertion graph on a first symbolic lattice domain; and

compute the symbolic simulation relation for the assertion graph on the first symbolic lattice domain to verify the assertion graph according to a normal satisfiability criteria.

9-13. (Canceled)

14. (Previously Presented) A computer implemented method for performing, at least in part, a formal verification of a circuit or other finite-state system, said method comprising:

initializing a symbolic simulation relation for an assertion graph on a first symbolic lattice domain, wherein the assertion graph on the first symbolic lattice domain is configured to express a justification property to verify through computing the symbolic simulation relation.

15. (Original) The method recited in Claim 14 further comprising:

computing the symbolic simulation relation for the assertion graph on the first symbolic lattice domain; and

checking the symbolic simulation relation to verify a plurality of properties expressed by a plurality of corresponding assertion graph instances, having at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

16. (Previously Presented) A computer implemented method for performing, at least in part, a formal verification of a circuit or other finite-state system, said method comprising:

specifying a justification property with an assertion graph.

17. (Original) The method recited in Claim 16 wherein the assertion graph is on a first symbolic lattice domain; and the justification property is expressed by one of a plurality of instances of the assertion graph, at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

18. (Original) The method recited in Claim 17 further comprising:

computing a symbolic simulation relation for the assertion graph on the first symbolic lattice domain; and

checking the symbolic simulation relation with a symbolic consequence labeling for the assertion graph on the first symbolic lattice domain according to a normal satisfiability criteria.

19-27. (Canceled)

28. (Previously Presented) A computer implemented verification system for performing, at least in part, a formal verification of a circuit or other finite-state system, said system comprising:

means for initializing a symbolic simulation relation for an assertion graph on a first symbolic lattice domain, wherein the assertion graph on the first symbolic lattice domain is configured to express a justification property to verify through computing the symbolic simulation relation;

means for computing the symbolic simulation relation for the assertion graph on the first symbolic lattice domain; and

means for checking the symbolic simulation relation to verify a plurality of properties expressed by a plurality of corresponding assertion graph instances, having at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

29-30. (Canceled)

31. (Previously Presented) The computer software product recited in Claim 4 wherein initializing the symbolic simulation relation comprises causing the processing device

to:

join a Boolean predicate for an outgoing edge from an initial vertex in the assertion graph with a symbolic antecedent labeling of an edge in the assertion graph.

32. (Previously Presented) The computer software product recited in Claim 31 wherein the symbolic antecedent labeling comprises a symbolic indexing function to encode a plurality of antecedent labels for a plurality of assertion graph instances, having at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

33. (Previously Presented) The computer software product recited in Claim 8 wherein the initialized symbolic simulation relation comprises a Boolean predicate for an outgoing edge from an initial vertex in the assertion graph joined with a symbolic antecedent labeling of an edge in the assertion graph.

34. (Previously Presented) The computer software product recited in Claim 33 wherein the symbolic antecedent labeling comprises a symbolic indexing function to encode a plurality of antecedent labels for a plurality of assertion graph instances, having at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

35. (Previously Presented) The computer software product recited in Claim 34 which, when executed by a processing device, further causes the processing device to:

check the symbolic simulation relation to verify a property expressed by at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

36. (Previously Presented) The computer software product recited in Claim 8 which, when executed by a processing device, further causes the processing device to:

check the symbolic simulation relation with a symbolic consequence labeling for the assertion graph on the first symbolic lattice domain according to a normal satisfiability criteria.

37. (Previously Presented) The computer software product recited in Claim 36 wherein the assertion graph is on a first symbolic lattice domain; and the justification property is expressed by one of a plurality of instances of the assertion graph, at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

38. (Previously Presented) The computer implemented verification system recited in Claim 28 wherein the symbolic simulation relation is initialized to comprise a Boolean predicate for an outgoing edge from an initial vertex in the assertion graph joined with a symbolic antecedent labeling of an edge in the assertion graph.

39. (Previously Presented) The computer implemented verification system recited in Claim 38 wherein the symbolic antecedent labeling comprises a symbolic indexing

function to encode a plurality of antecedent labels for a plurality of assertion graph instances, having at least one assertion graph instance on a second lattice domain different from the first symbolic lattice domain.

40. (Previously Presented) The computer implemented verification system recited in Claim 28 wherein the symbolic simulation relation is checked with a symbolic consequence labeling for the assertion graph on the first symbolic lattice domain according to a normal satisfiability criteria.

IX. Evidence Appendix: With Copies of Evidence Relied Upon by Appellant

Appellant relies upon no additional evidence in this appeal.

X. Related Proceedings Appendix: Copies of Decisions Rendered by a Court or the Board in any Prior and Pending Appeals, Interferences or Judicial Proceedings

There are no related appeals or interferences to appellant's knowledge that would have a bearing on any decision of the Board of Patent Appeals and Interferences.